Neural Harmonic-plus-Noise Waveform Model with Trainable Maximum Voice Frequency for Text-to-Speech Synthesis

Neural Harmonic-plus-Noise Waveform Model with Trainable Maximum Voice Frequency for Text-to-Speech Synthesis

Xin Wang[1]*, Junichi Yamagishi*[12]

[1]National Institute of Informatics, Japan, [2]CSTR, University of Edinburgh, UK

wangxin@nii.ac.jp, jyamagis@nii.ac.jp

Abstract

Neural source-filter (NSF) models are deep neural networks that produce waveforms given input acoustic features. They use dilated-convolution-based neural filter modules to filter a sine-based excitation for waveform generation, which is different from WaveNet and flow-based models. One of the NSF models, called harmonic-plus-noise NSF (h-NSF) model, uses separate pairs of source and neural filter to generate harmonic and noise waveform components. It performed as well as WaveNet in terms of speech quality while being superior in generation speed. However, h-NSF may be further improved. While h-NSF merges the harmonic and noise components using pre-defined digital low- and high-pass filters, it is well known that the maximum voice frequency (MVF) that separates the periodic and aperiodic spectral bands are time variant. Therefore, we propose a new h-NSF model with time-variant and trainable MVF. We parameterize the digital low- and high-pass filters as windowed-sinc filters and predict their cut-off-frequency (i.e., MVF) from input acoustic features. Our experiments demonstrated that the new model can predict a good trajectory of MVF. The quality of the generated speech was slightly improved, and the fast generation speed was maintained.

Index Terms: speech synthesis, source-filter model, harmonic-pluse-noise waveform model, neural network

1. Introduction

In text-to-speech (TTS) systems using statistical parametric speech synthesis (SPSS) [1], neural-network (NN)-based models have been introduced to both the front-end text analyzer and the back-end acoustic models [2, 3, 4, 5, 6]. A recent trend is to replace the signal-processing-based vocoder with a neural waveform model, a component that generates an waveform given the acoustic features predicted by the acoustic models.

One well known neural waveform model called WaveNet-vocoder [7] uses a dilated convolution (CONV) network [8] to produce the waveform samples in an autoregressive (AR) manner, i.e., generating the current waveform sample with the previously generated samples as condition. Although WaveNet outperformed traditional vocoders [9], its sequential generation process is prohibitively slow. Flow-based models [10, 11, 12] convert a noise sequence into a waveform in one shot. However, some of them requires sequential processing during training [10], which dramatically increases the training time [13]. Others use knowledge distilling to transfer the knowledge from an AR WaveNet to a flow-based student model, which is complicated in implementation.

We recently proposed neural source-filter (NSF) waveform models, which requires neither AR, knowledge distilling, nor flow-based methods [14]. The NSF models generally use three modules to generate a waveform: a conditional module that upsamples input acoustic features such as F0 and Mel-spectrograms, a source module that outputs a sine-based excitation given the F0, and a filter module that uses dilated-CONV blocks to morph the excitation into a waveform. The models are trained to minimize the spectral amplitude distances between the generated and natural waveforms. Without the flow-based approaches, the NSF models are easy to implement and train. Without the AR structure, the NSF models are at least 100 times faster than WaveNet in waveform generation [15].

One NSF model called harmonic-plus-noise NSF (h-NSF) inherits the efficiency of the NSF models and demonstrated equally well or better performance than WaveNet and other NSF models on a Japanese dataset [15]. The core idea of h-NSF is to use separate pairs of source and neural filter modules to generate a harmonic and a noise waveform component before merging the two components into an output waveform by using pre-defined finite impulse response (FIR) filters. The harmonic-plus-noise architecture of h-NSF improves the quality of generated waveforms, especially on unvoiced sounds.

It is well known that the speech spectrum can be roughly divided into a periodic and an aperiodic bands by a maximum voice frequency (MVF) [16]. Although MVF is time variant, our h-NSF chooses one of the two pre-defined MVF values (i.e., the cut-off frequency of FIR filters) according to the voicing status of the sound. In this paper, we propose a new h-NSF model with trainable MVF. This new model parameterizes the FIR filters as windowed-sinc filters [17] and predict their MVF values from the input acoustic features. Experiments demonstrated that the new h-NSF can predict the MVF reasonably well on the basis of the voicing status. The quality of generated waveforms were also improved and the generation speed was maintained.

Because the new h-NSF model replies on windowed-sinc filters, we refer to it as *sinc-h-NSF* while the previous h-NSF model as *base-h-NSF*. In Section 2, we will explain the base-h-NSF in details. In Section 3, we explain the sinc-h-NSF. In Section 4, we compare the two h-NSF models with WaveNet in experiments. In Section 5, we draw a conclusion.

2. Review of base-h-NSF model

A neural waveform model converts input acoustic features into an output waveform. Let us denote the input acoustic feature sequence as $\boldsymbol{c}_{1:B} = \{\boldsymbol{c}_1, \cdots \boldsymbol{c}_B\}$, where $\boldsymbol{c}_b \in \mathbb{R}^D$ is the feature vector for the b-th frame. We then use $\boldsymbol{o}_{1:T} = \{o_1, \cdots, o_T\}$ and $\widehat{\boldsymbol{o}}_{1:T}$ to denote the natural and generated waveforms, respectively. Here, T is the waveform length and $o_t \in \mathbb{R}$ is the waveform value at the t-th sampling point.

In our previous work, we proposed NSF models [14] to convert $\boldsymbol{c}_{1:B}$ into $\widehat{\boldsymbol{o}}_{1:T}$. The NSF models use three types of modules: a source to produce an excitation signal, a neural filter to convert the excitation into $\widehat{\boldsymbol{o}}_{1:T}$, and a condition part to processes input $\boldsymbol{c}_{1:B}$ for the other two modules. The training

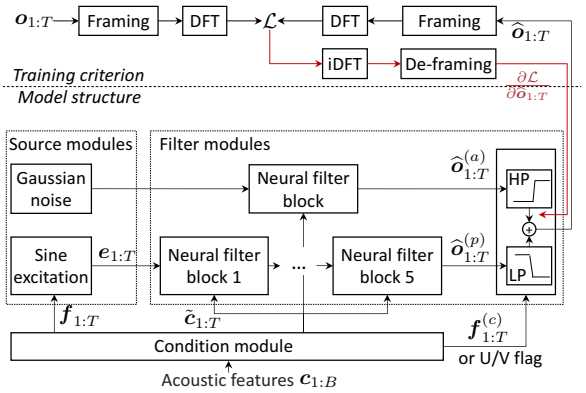

Figure 1: *General network structures for baseline and proposed trainable h-NSF models. LP and HP denote low- and high-pass FIR filters respectively. Red arrows denote gradients.*

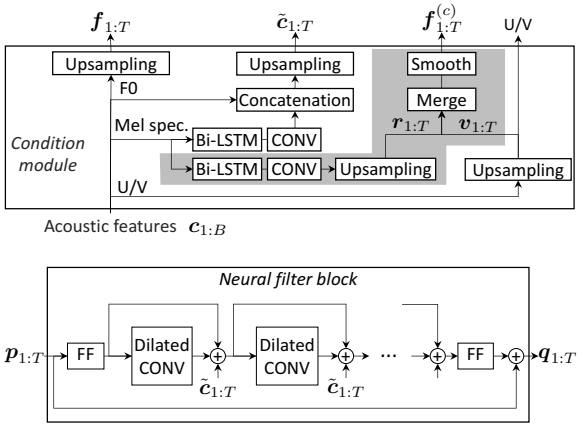

Figure 2: *Condition module (top) and neural filter block (bottom). FF, Bi-LSTM and CONV denote feedforward, bi-directional LSTM and convolutional layers, respectively. Layers in shaded area are only used to compute $\boldsymbol{f}_{1:T}^{(c)}$ for the proposed new h-NSF model.*

is conducted by minimizing the spectral distance between $\widehat{\boldsymbol{o}}_{1:T}$ and $\boldsymbol{o}_{1:T}$ [14]. Base-h-NSF model introduces a harmonic-plus-noise architecture to the NSF framework as Figure 1 plots. The details of base-h-NSF are explained in the following sections.

2.1. Condition module

The condition module is the bedrock of base-h-NSF. Its basic task is to upsample the frame-rate acoustic features to the waveform rate. As Figure 1 shows, the condition module processes three types of features[1]: the upsampled F0 sequence $\boldsymbol{f}_{1:T}$ for the source module, the upsampled and transformed acoustic feature sequence $\widehat{\boldsymbol{c}}_{1:T}$ for the neural filter module, and the upsampled unvoiced/voiced (U/V) flag for the FIR filters.

Suppose each frame of the input $\boldsymbol{c}_{1:B}$ contains an F0 datum $f_b \in \mathbb{R}_{\geq 0}$ and a Mel-spectrum $\boldsymbol{s}_b$, i.e., $\boldsymbol{c}_b = [f_b, \boldsymbol{s}_b^\top]^\top$. Then, it is straightforward to upsample the F0 sequence $\{f_1, \cdots, f_B\}$ of length B into $\boldsymbol{f}_{1:T}$ of length T by simply copying each f_b for $\lceil T/B \rceil$ times. Similarly, the U/V flag sequence then can be upsampled after determining the U/V from the f_b (e.g., voiced if $f_b > 0$ or unvoiced if $f_b = 0$). For $\widehat{\boldsymbol{c}}_{1:T}$, the condition module first transforms the sequence of $\boldsymbol{s}_b$ using two hidden layers: a bi-directional LSTM (Bi-LSTM) layer with a layer size of 64 and a 1-D convolutional (CONV) layer with a layer size of 63 and a window size of 3. After that, it concatenates the output feature vector with the F0 and upsamples the vector as $\tilde{\boldsymbol{c}}_{1:T}$, where $\tilde{\boldsymbol{c}}_t \in \mathbb{R}^{64}, \forall t \in \{1, \cdots, T\}$.

2.2. Source modules

The base-h-NSF model contains two source modules. One module generates a Gaussian noise excitation for the noise waveform component, while the other that generates a sine-based excitation signal $\boldsymbol{e}_{1:T}$ for the harmonic component.

Here we briefly explain the sine-based excitation. Given the upsampled F0 sequence $\boldsymbol{f}_{1:T}$, a sine waveform that carries the F0 or the i-th harmonic can be generated as

$$e_t^{<i>} = \begin{cases} \alpha \sin(\sum_{k=1}^{t} 2\pi \frac{i f_k}{N_s} + \phi) + n_t, & \text{if } f_t > 0 \\ \frac{\alpha}{3\sigma} n_t, & \text{if } f_t = 0 \end{cases}, \quad (1)$$

[1] There is one alternative feature $\boldsymbol{f}_{1:T}^{(c)}$ that is used by the trainable h-NSF model proposed in this paper (Section 3). It is not used by the baseline h-NSF models.

where $\phi \in [-\pi, \pi]$ is a random initial phase, N_s is the waveform sampling rate, and $n_t \sim \mathcal{N}(0, \sigma^2)$ is a Gaussian noise. Note that $e_t^{<i>}$ is a Gaussian noise in unvoiced regions where $f_t = 0$. The hyper-parameter α adjusts the amplitude of $e_t^{<i>}$, while σ is the standard deviation of the Gaussian distribution. We set $\sigma = 0.003$ and $\alpha = 0.1$ [15].

We set $I = 8$ for base-h-NSF, i.e., using fundamental tone and 7 higher harmonics. Given these 8 signals, a feedforward (FF) layer is used to merge them into the excitation $\boldsymbol{e}_{1:T} = \tanh(\sum_{i=1}^{I} w_i \boldsymbol{e}_{1:T}^{<i>} + w_b)$, where $e_t \in \mathbb{R}, \forall t \in \{1, \cdots, T\}$. Note that $\{w_1, \cdots w_I, w_b\}$ are the FF layer's weights.

2.3. Filter modules

The filter modules of the base-h-NSF can be described in three parts. The first part uses one neural filter block to convert the Gaussian noise into the noise waveform component $\widehat{\boldsymbol{o}}_{1:T}^{(a)}$, while the second part uses five blocks to convert $\boldsymbol{e}_{1:T}$ into the harmonic waveform component $\widehat{\boldsymbol{o}}_{1:T}^{(p)}$. The third part uses FIR filters to merge $\widehat{\boldsymbol{o}}_{1:T}^{(a)}$ and $\widehat{\boldsymbol{o}}_{1:T}^{(p)}$ into the output waveform $\widehat{\boldsymbol{o}}_{1:T}$.

The neural filter block is plotted in Figure 2. Suppose the input signal is $\boldsymbol{p}_{1:T}$, where $p_t \in \mathbb{R}, \forall t \in \{1, \cdots, T\}$[2]. Each $\boldsymbol{p}_t$ is first expanded to 64 dimensions through an FF layer, then processed by a dilated-CONV layer with 64 output channels, and finally added with the output of the dilated-CONV layer and the conditional feature $\tilde{\boldsymbol{c}}_{1:T}$. This process is repeated for 10 times, and the final output sequence is transformed back into a 1-dimensional signal through a FF layer and then summed with $\boldsymbol{p}_{1:T}$. Note that the dilation size of the k-th dilated-CONV layer is $2^{\text{mod(k-1, 10)}}$, and its filter size is set to 3.

After the neural filter blocks generate $\widehat{\boldsymbol{o}}_{1:T}^{(p)}$ and $\widehat{\boldsymbol{o}}_{1:T}^{(a)}$, the base-h-NSF uses low- and high-pass FIR filters to mix them as the output waveform $\widehat{\boldsymbol{o}}_{1:T} = \text{Low-pass}(\widehat{\boldsymbol{o}}_{1:T}^{(p)}) + \text{High-pass}(\widehat{\boldsymbol{o}}_{1:T}^{(a)})$. In implementation, we switch the cut-off frequency (-3 dB) of the FIR filters given the U/V flag. In voiced regions, the cut-off frequency values for the low- and high-pass filters are 5 kHz and 7 kHz, respectively. In unvoiced regions, they are 1 kHz and 3kHz. The filter coefficients are calculated in advance [18] and fixed in the model.

[2] For the 1st block that receives $\boldsymbol{e}_{1:T}$ as input, $\boldsymbol{p}_{1:T} = \boldsymbol{e}_{1:T}$.

3. Proposed h-NSF model with trainable maximum voice frequency

The cut-off frequency of the FIR filters in base-h-NSF is specified by expert knowledge and only changes according to voicing condition. In the classical harmonic-plus-noise models, however, the cut-off frequency is assumed to be time variant [16, 19]. It is thus reasonable to try time-variant FIR filters with the cut-off frequency predicted from the input acoustic features.

The proposed h-NSF model is identical to the base-h-NSF except the procedure to calculate the time-variant cut-off-frequency for the FIR filters. Suppose we are using filters of order M, and their coefficients at time t are $\boldsymbol{h}_t^{(p)} = \{h_{t,0}^{(p)}, \cdots, h_{t,M}^{(p)}\}$, and $\boldsymbol{h}_t^{(a)} = \{h_{t,0}^{(a)}, \cdots, h_{t,M}^{(a)}\}$, respectively. Given the periodic and aperiodic components $\{\widehat{\boldsymbol{o}}_{1:T}^{(p)}, \widehat{\boldsymbol{o}}_{1:T}^{(a)}\}$, the output waveform $\widehat{o}_t$ at the t-th time step can merged as

$$\widehat{o}_t = \sum_{m=0}^{M-1} \widehat{o}_{t-m}^{(p)} h_{t,m}^{(p)} + \sum_{m=0}^{M-1} \widehat{o}_{t-m}^{(a)} h_{t,m}^{(a)}. \tag{2}$$

Our goal is to predict $\{\boldsymbol{h}_t^{(p)}, \boldsymbol{h}_t^{(a)}\}$ from the acoustic features $\boldsymbol{c}_{1:B}$. For this purpose, we use a two-step procedure as Figure 3 plots. First, the condition module predicts a normalized cut-off frequency $f_t^{(c)} \in (0, 1)$[3] given $\boldsymbol{c}_{1:B}$. After that, $\{\boldsymbol{h}_t^{(p)}, \boldsymbol{h}_t^{(a)}\}$ are calculated from $f_t^{(c)}$. During back propagation, the gradients are computed and propagated backwards.

3.1. Forward computation

3.1.1. Predicting cut-off frequency

Because the MVF of a sound is influenced by its voicing status, we take the U/V flag into consideration and revise the condition module of the base-h-NSF in order to predict $f_t^{(c)}$ from $\boldsymbol{c}_{1:B}$. As the shaded area of Figure 2 plots, a Bi-LSTM layer and a CONV layer with a tanh activation function are added to predict a signal that will be upsampled to $\boldsymbol{r}_{1:T}$, where $r_t \in (-1, 1), \forall t \in \{1, \cdots, T\}$. Meanwhile, the U/V flag is upsampled as the signal $\boldsymbol{v}_{1:T}$. We then set $v_t = 0.7$ if the t-th time step is voiced or $v_t = 0.3$ for an unvoiced time step[4].

With $v_t \in \{0.7, 0.3\}$ and $r_t \in (-1, 1)$, we can fuse them into the output $f_t^{(c)} \in (0, 1)$ in various ways. Without loss of generality, we design a fusion function as

$$f_t^{(c)} = \mathcal{F}(av_t + br_t + c), \tag{3}$$

where $\{a, b, c\}$ can be trainable parameters or fixed hyper-parameters. $\mathcal{F}(\cdot)$ can be a sigmoid function or an identity function $f(x) = x$ if $av_t + br_t + c$ is already between 0 and 1.

How we define Equation (3) depends on our prior knowledge about the MVF and its relationship with the voicing status. For example, we may use three definitions listed in Table 1. The first definition ensures that $f_t^{(c)} \in (0.1, 0.5)$ in voiced time steps while $f_t^{(c)} \in (0.5, 0.9)$ in unvoiced time steps. The second definition only replies on the r_t, i.e., predicting MVF from conditional features without any prior knowledge of voicing status. Finally, the last definition learns the weight to combine the v_t and r_t. We will compare the three definitions in experiments. Note that, to prevent abrupt change of the filter frequency response, $\boldsymbol{f}_{1:T}^{(c)}$ is smoothed by taking the time domain average over a window size of 5 ms.

[3]Being normalized means that $f_t^{(c)}$ is equal to the physical cut-off frequency (Hz) divided by the Nyquist frequency.

[4]These values are references suggesting that, for example, the MVF of voiced sounds is around 5.6 kHz (=0.7 * 8 kHz). We can scale or shift these values when we merge v_t with r_t.

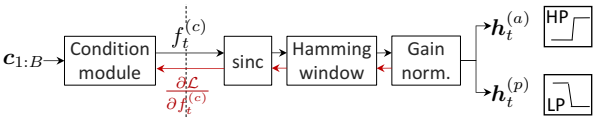

Figure 3: *Procedure to derive low-pass (LP) and high-pass (HP) filter coefficients in proposed trainable h-NSF model.*

Table 1: *Four possible definitions of* $f_t^{(c)} = \mathcal{F}(av_t + br_t + c)$ *(Equation (3)) to merge* $v_t \in \{0.7, 0.3\}$ *and* $r_t \in (-1, 1)$.

Definition	$\mathcal{F}(x)$	a	b	c
$f_t^{(c)} = v_t + 0.2r_t$	x	1	0.2	0
$f_t^{(c)} = 0.5r_t + 0.5$	x	0	0.5	0.5
$f_t^{(c)} = \mathcal{F}(av_t + br_t + c)$	sigmoid	trainable		

3.1.2. Windowed-sinc filters

Given $f_t^{(c)}$, we follow the standard procedure to design windowed-sinc filters and calculate $\{\boldsymbol{h}_t^{(p)}, \boldsymbol{h}_t^{(a)}\}$. Suppose the filter length M is an odd number, and the index of the filter coefficients is centered around 0, i.e., $n \in \{-\frac{M-1}{2}, -\frac{M-1}{2} + 1, \cdots, 0, \cdots, \frac{M-1}{2}\}$. Given the $f_t^{(c)}$ at the t-th time step, a coefficient can be computed for each n based on the sinc function and the Hamming window Hamm$(\cdot)$:

$$\begin{aligned} \tilde{h}_{t,n}^{(p)} &= f_t^{(c)} \text{sinc}(\pi f_t^{(c)} n) \text{Hamm(n)} \\ &= \frac{\sin(\pi f_t^{(c)} n)}{\pi n} \big(0.54 + 0.46 \cos(\frac{2\pi n}{M})\big). \end{aligned} \tag{4}$$

The desired filter coefficients $\boldsymbol{h}_t^{(p)}$ can then be calculated after a gain normalization and index shifting from n to m:

$$h_{t,m}^{(p)} = \frac{\tilde{h}_{t,m-\frac{M-1}{2}}^{(p)}}{\sum_{n=-\frac{M-1}{2}}^{\frac{M-1}{2}} \tilde{h}_{t,n}^{(p)}} \tag{5}$$

Note that the gain normalization makes the low-pass filter's gain equal to 1 at 0 Hz. The index shifts the index from $n \in \{-\frac{M-1}{2}, \cdots, 0, \cdots, \frac{M-1}{2}\}$ to $m \in \{0, \cdots, M\}$ to make the filter causal. Similarly, the high-pass filter coefficients are deterministically computed by:

$$\tilde{h}_{t,n}^{(a)} = \Big(\frac{\sin(\pi n)}{\pi n} - \frac{\sin(\pi f_t^{(c)} n)}{\pi n}\Big) \text{Hamm}(n), \tag{6}$$

$$h_{t,m}^{(a)} = \frac{\tilde{h}_{t,m-\frac{M-1}{2}}^{(a)}}{\sum_{n=-\frac{M-1}{2}}^{\frac{M-1}{2}} \tilde{h}_{t,n}^{(a)} (-1)^n}. \tag{7}$$

Trainable sinc-based FIR filters have been used in SincNet [20]. While SincNet uses multiple time invariant band-pass filters, we used time variant low- and high-pass ones. The cut-off frequency in SincNet is assumed to be the parameter of the network, but our network predicts it from conditional features.

3.2. Back propagation

We need to calculate the gradients of $f_t^{(c)}$ w.r.t the loss function $\mathcal{L}$. By using the chain rule on Equation (2), we first get

$$\frac{\partial \mathcal{L}}{\partial f_t^{(c)}} = \frac{\partial \mathcal{L}}{\partial \hat{o}_t} \frac{\partial \hat{o}_t}{\partial f_t^{(c)}} = \frac{\partial \mathcal{L}}{\partial \hat{o}_t} \sum_{m=0}^{M-1} (\hat{o}_{t-m}^{(p)} \frac{\partial h_{t,m}^{(p)}}{\partial f_t^{(c)}} + \hat{o}_{t-m}^{(a)} \frac{\partial h_{t,m}^{(a)}}{\partial f_t^{(c)}}). \tag{8}$$

Then, based on Equations (4) and (5), it can be shown that

$$\frac{\partial h_{t,m}^{(p)}}{\partial f_t^{(c)}} = \sum_n \frac{\partial h_{t,m}^{(p)}}{\partial \tilde{h}_{t,n}^{(p)}} \frac{\partial \tilde{h}_{t,n}^{(p)}}{\partial f_t^{(c)}} = \frac{\alpha_{t,m-\frac{M-1}{2}} - h_{t,m}^{(p)} \gamma_t^{(p)}}{\beta_t^{(p)}}, \tag{9}$$

where $\alpha_{t,n} = \text{Hamm}(n) \cos(\pi f_t^{(c)} n)$, $\beta_t^{(p)} = \sum_n \tilde{h}_{t,n}^{(p)}$, and $\gamma_t^{(p)} = \sum_n \alpha_{t,n}$. Similarly, $\partial h_{t,m}^{(a)} / \partial f_t^{(c)}$ can be calculated as

$$\frac{\partial h_{t,m}^{(a)}}{\partial f_t^{(c)}} = \frac{h_{t,m}^{(a)} \gamma_t^{(a)} - \alpha_{t,m-\frac{M-1}{2}}}{\beta_t^{(a)}}, \tag{10}$$

where $\beta_t^{(a)} = \sum_n (-1)^{(n)} \tilde{h}_{t,n}^{(a)}$ and $\gamma_t^{(a)} = \sum_n (-1)^n \alpha_{t,n}$.

On the basis of Equations (9) and (10), $\partial \mathcal{L} / \partial f_t^{(c)}$ in Equation (8) can be computed and propagated backwards.

4. Experiments

4.1. Data and feature configuration

The experiment in this paper used the same data corpus and feature configuration as our previous work [15]. Specifically, the corpus is a neural-style reading speech dataset from a Japanese female speaker. The original speech waveforms were down sampled from 48 kHz to 16 kHz for experiments.

To train the neural waveform models, we randomly selected 9,000 utterances (15 hours) as the training set. We then prepared a validation set with 500 randomly selected utterances and a test set with another 480 utterances. The acoustic features included the Mel-spectrograms of 80 dimensions and the F0 extracted using an ensemble of pitch estimators [21]. The frame shift of the acoustic features is 5 ms (200 Hz).

Because we planned to evaluate the neural waveform models not only in copy-synthesis but also in TTS scenarios, we also extracted linguistic features from the transcripts to train acoustic models that predict the Mel-spectrogram and F0 from the text. The linguistic features contained quin-phone identity, phrase accent type, etc [22]. These features were then aligned against the acoustic features sequences.

4.2. Experimental models

We compared the following models in the experiment:

- `WaveNet`: an AR WaveNet;
- `base-h-NSF`: base-h-NSF using the fixed coefficients for the low- and high-pass filters;
- `sinc1-h-NSF`: h-NSF with windowed-sinc filters and cut-off frequency $f_t^{(c)} = v_t + 0.2 r_t$;
- `sinc2-h-NSF`: h-NSF with windowed-sinc filters and cut-off frequency $f_t^{(c)} = 0.5 r_t + 0.5$;
- `sinc3-h-NSF`: h-NSF with windowed-sinc filters and cut-off frequency $f_t^{(c)} = \text{Sigmoid}(a v_t + b r_t + c)$;

Table 2: *Short time analysis configurations for the spectral amplitude distance of NSF models*

	$\mathcal{L}_1$	$\mathcal{L}_2$	$\mathcal{L}_3$
DFT bins	512	128	2048
Frame length	320 (20 ms)	80 (5 ms)	1920 (120 ms)
Frame shift	80 (5 ms)	40 (2.5 ms)	640 (40 ms)

Note: all configurations use Hann window.

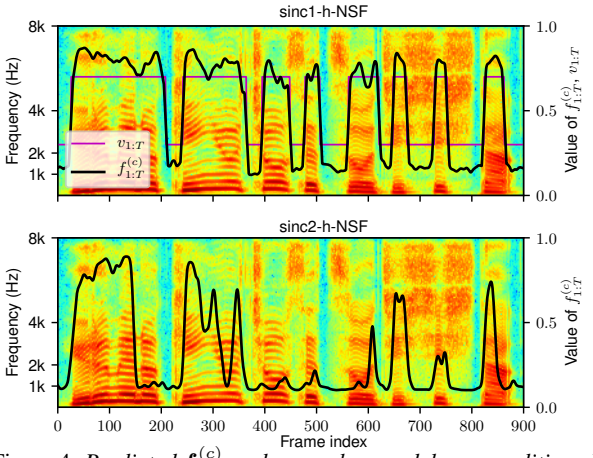

Figure 4: *Predicted $\boldsymbol{f}_{1:T}^{(c)}$ and $\boldsymbol{v}_{1:T}$ when model was conditioned on natural acoustic features. Background is the spectrogram of natural waveform. Figure for `sinc3-h-NSF` is not plotted because the generated $\boldsymbol{f}_{1:T}^{(c)}$ was 1.0.*

`base-h-NSF` was trained in our previous work [15]. It used 5 dilated-CONV filter blocks (Figure 2) to generate the harmonic waveform component, and each block contained 10 dilated-CONV layers. The k-th dilated-CONV layer had a dilation size of 2^{k-1}. For the noise component, `base-h-NSF` only used one block. The three `sinc*-h-NSF` models used the same network structure as `base-h-NSF` except the hidden layers to predict cut-off frequency for the time-variant FIR filters. The FIR filters used $M = 31$. All the NSF models were trained using the sum of three spectral amplitude distances with framing and windowing configurations listed in Table 2.

`WaveNet` was trained in our another work [9]. It contained 40 dilated CONV layers, where the k-layer had a dilation size of $2^{\text{mod}(k-1,10)}$. `WaveNet` took both Mel-spectrogram and F0 as conditional features and generated 10-bit μ-law quantized waveform values. Note that the number of parameters `WaveNet` was more than that of the h-NSF models [15].

To predict the acoustic features from the linguistic features, we used a deep neural AR F0 model [6] to predict the F0 and another deep AR model for the Mel-spectrogram. The acoustic feature sequences were generated given the duration aligned on the test set waveforms.

4.3. Results and analysis

We first compare the predicted MVF from the `sinc*-h-NSF` models. Figure 4 plots the predicted MVF trajectory and the natural waveform spectrogram. Without using u/v, `sinc2-h-NSF` failed to predict MVF for some voiced regions, for example from the 400-th to 500-th frames. Although `sinc3-h-NSF` used the u/v, the function $\text{Sigmoid}(a v_t + b r_t + c)$ was saturated and produced 1.0 for all the time steps. It seemed to be difficult to learn a trainable function to merge the

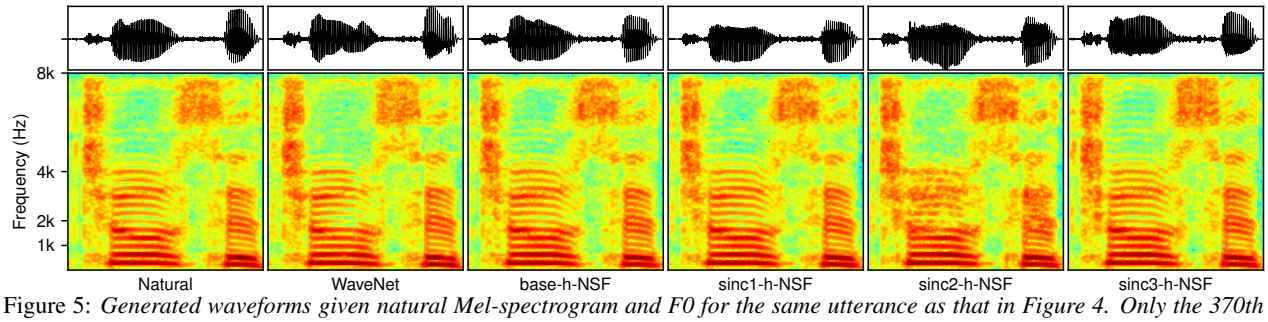

Figure 5: *Generated waveforms given natural Mel-spectrogram and F0 for the same utterance as that in Figure 4. Only the 370th to 510th frames are plotted.*

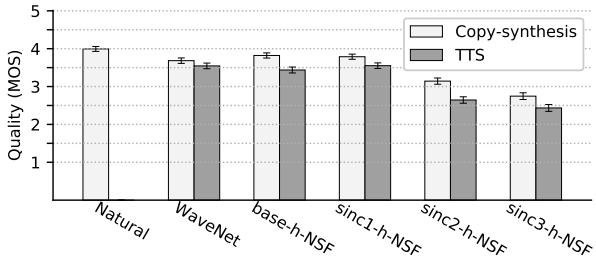

Figure 6: *MOS scores of experimental models. Error bars at confidence level of 95% are plotted.*

u/v and the other acoustic features for MVF prediction. MVF predicted from `sinc1-h-NSF` is roughly consistent with the spectrogram, i.e., a high MVF in voiced regions and a low MVF in unvoiced regions. These results suggest that MVF can be predicted reasonably well by summing the u/v with a residual signal predicted from input acoustic features.

We then compared the quality of the generated waveforms from the experimental models in a subjective evaluation test. In a single evaluation round, an evaluator listened to one speech waveform file on one screen, rated the speech quality on a 1-to-5 MOS scale, and repeated the process for multiple screens. The waveforms in one evaluation round were for the same text and were played in a random order. Each evaluator can replay the waveform file during the evaluation. All the waveforms were converted to 16-bit PCM format in advance.

Around 150 evaluators participated in this test, and 1604 sets of MOS scores were obtained. The results plotted in Figure 6 demonstrated that `sinc1-h-NSF`, `base-h-NSF`, and `WaveNet` performed equally well. In contrast, `sinc2-h-NSF` and `sinc3-h-NSF` lagged behind. The reason for `sinc2-h-NSF`'s poor performance is the 'under-estimated' MVF in voiced regions as Figure 4 shows. As the result, some voiced sounds generated by `sinc2-h-NSF` were over-aperiodic. For example, as Figure 5 plots, the voiced sound only had a weak harmonic structure around the 4 kHz. `sinc3-h-NSF` generated $\boldsymbol{f}_{1:T}^{(c)} = 1$ for all utterances and the waveforms generated from `sinc3-h-NSF` lacked aperiodicity, which can be observed from Figure 5. Furthermore, unvoiced sounds such as [s] was less aperiodic (see Figure 7) and sounded like a pulse train.

Finally, Table 3 shows the number of parameters and generation speed. `WaveNet` was slow because of the AR generation process. However, the NSF models were much faster because they produced the waveform in one shot. In the memory-save mode, in which the NSF-models reduce GPU memory consumption by releasing and allocating memory layer by layer, the generation speed decreased because of the time for memory operation. However, they still surpassed `WaveNet`.

Table 3: *Number of network parameters and average number of waveform samples generated in 1-sec time on single Nvidia P100 GPU card.* `sinc*-h-NSF` *had similar performance.*

Model	No. of model parameters	Generation speed	
		memory-save	normal mode
`WaveNet`	$2.96e+6$	-	0.19 k
`base-h-NSF`	$1.20e+6$	71 k	335 k
`sinc1-h-NSF`	$1.20e+6$	70 k	335 k

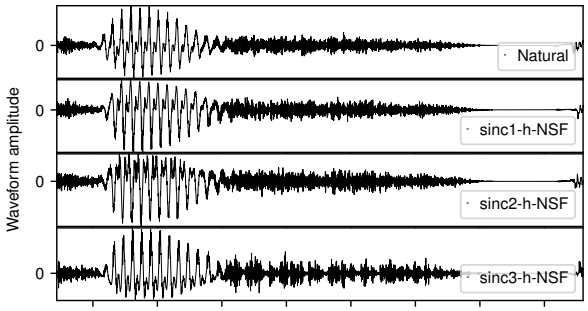

Figure 7: *Natural and generated waveform from models given natural acoustic features.*

5. Conclusions

We proposed a new h-NSF model with trainable MVF. Compared with the baseline h-NSF model using pre-defined FIR filters to merge the harmonic and noise waveform components, the new h-NSF model predicts a time-variant MVF from the input acoustic features to adjust the frequency response of the FIR filters. We compared different strategies to predict the MVF in the experiments and found that U/V information can be useful prior knowledge. Specifically, we can predict a residual signal from the input acoustic features and add it to the U/V signal, which was more stable than other strategies such as directly predicting the MVF from scratch. Experiment demonstrated that the proposed trainable h-NSF can generate high-quality waveforms as well as WaveNet. Meanwhile, the waveform generation speed was roughly comparable to other NSF models nad was much faster than WaveNet.

Acknowledgement: This work was partially supported by a JST CREST Grant (JPMJCR18A6, VoicePersonae project), Japan, and MEXT KAKENHI Grants (16H06302, 17H04687, 18H04120, 18H04112, 18KT0051), Japan. The experiments partially were conducted using TSUBAME 3.0 supercomputer of Tokyo Institute of Technology.

6. References

[1] H. Zen, K. Tokuda, and A. W. Black, "Statistical parametric speech synthesis," *Speech Communication*, vol. 51, pp. 1039–1064, 2009.

[2] K. Yao and G. Zweig, "Sequence-to-sequence neural net models for grapheme-to-phoneme conversion," in *Proc. Interspeech*, 2015, pp. 3330–3334.

[3] H. Zen, A. Senior, and M. Schuster, "Statistical parametric speech synthesis using deep neural networks," in *Proc. ICASSP*, 2013, pp. 7962–7966.

[4] T. Kaneko, H. Kameoka, N. Hojo, Y. Ijima, K. Hiramatsu, and K. Kashino, "Generative adversarial network-based postfilter for statistical parametric speech synthesis," in *Proc. ICASSP*, 2017, pp. 4910–4914.

[5] G. E. Henter, S. Ronanki, O. Watts, M. Wester, Z. Wu, and S. King, "Robust TTS duration modelling using DNNs," in *Proc. ICASSP*, 2016, pp. 5130–5134.

[6] X. Wang, S. Takaki, and J. Yamagishi, "Autoregressive neural f0 model for statistical parametric speech synthesis," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 8, pp. 1406–1419, 2018.

[7] A. Tamamori, T. Hayashi, K. Kobayashi, K. Takeda, and T. Toda, "Speaker-dependent WaveNet vocoder," in *Proc. Interspeech*, 2017, pp. 1118–1122.

[8] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "WaveNet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016.

[9] X. Wang, J. Lorenzo-Trueba, S. Takaki, L. Juvela, and J. Yamagishi, "A comparison of recent waveform generation and acoustic modeling methods for neural-network-based speech synthesis," in *Proc. ICASSP*, 2018, pp. 4804–4808.

[10] R. Prenger, R. Valle, and B. Catanzaro, "Waveglow: A flow-based generative network for speech synthesis," *arXiv preprint arXiv:1811.00002*, 2018.

[11] A. van den Oord, Y. Li, I. Babuschkin, K. Simonyan, O. Vinyals, K. Kavukcuoglu, G. van den Driessche, E. Lockhart, L. Cobo, F. Stimberg, N. Casagrande, D. Grewe, S. Noury, S. Dieleman, E. Elsen, N. Kalchbrenner, H. Zen, A. Graves, H. King, T. Walters, D. Belov, and D. Hassabis, "Parallel WaveNet: Fast high-fidelity speech synthesis," in *Proc. ICML*, 2018, pp. 3918–3926.

[12] W. Ping, K. Peng, and J. Chen, "Clarinet: Parallel wave generation in end-to-end text-to-speech," in *Proc. ICLP*, 2019.

[13] T. Okamoto, T. Toda, Y. Shiga, and H. Kawai, "Investigations of real-time neural vocoders with fundamental frequency and mel-cepstra," in *Proc. ASJ spring meeting*, 2019, p. (in Japanese).

[14] X. Wang, S. Takaki, and J. Yamagishi, "Neural source-filter-based waveform model for statistical parametric speech synthesis," in *Proc. ICASSP*, 2019, p. (to appear).

[15] ——, "Neural source-filter waveform models for statistical parametric speech synthesis," *arXiv:1904.12088*, 2019.

[16] Y. Stylianou, "Harmonic plus noise models for speech, combined with statistical methods, for speech and speaker modification," Ph.D. dissertation, Ecole Nationale Superieure des Telecommunications, 1996.

[17] S. W. Smith, *The Scientist and Engineer's Guide to Digital Signal Processing*. San Diego, CA, USA: California Technical Publishing, 1997.

[18] T. Parks and J. McClellan, "Chebyshev approximation for nonrecursive digital filters with linear phase," *IEEE Transactions on Circuit Theory*, vol. 19, no. 2, pp. 189–194, 1972.

[19] T. Drugman and Y. Stylianou, "Maximum voiced frequency estimation: Exploiting amplitude and phase spectra," *IEEE Signal Processing Letters*, vol. 21, no. 10, pp. 1230–1234, 2014.

[20] M. Ravanelli and Y. Bengio, "Speaker recognition from raw waveform with sincnet," in *2018 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2018, pp. 1021–1028.

[21] L. Juvela, X. Wang, S. Takaki, S. Kim, M. Airaksinen, and J. Yamagishi, "The NII speech synthesis entry for Blizzard Challenge 2016," in *Proc. Blizzard Challenge Workshop*, 2016.

[22] H.-T. Luong, X. Wang, J. Yamagishi, and N. Nishizawa, "Investigating accuracy of pitch-accent annotations in neural-network-based speech synthesis and denoising effects," in *Proc. Interspeech*, 2018, pp. 37–41.